


```

LL      IIIIII  BBBB8888  GGGGGGGG  EEEEEEEEE  TTTTTTTTTT  IIIIII  NN      NN  PPPPPPPP
LL      IIIIII  88888888  GGGGGGGG  EEEEEEEEE  TTTTTTTTTT  IIIIII  NN      NN  PPPPPPPP
LL      II      88      88  GG      GG  EE      EE  TT      TT  II      II  NN      NN  PP      PP
LL      II      88      88  GG      GG  EE      EE  TT      TT  II      II  NN      NN  PP      PP
LL      II      88      88  GG      GG  EE      EE  TT      TT  II      II  NN      NN  PP      PP
LL      II      88888888  GG      GG  EEEEEEEEE  TT      TT  II      II  NNNN  NN  PP      PP
LL      II      88888888  GG      GG  EEEEEEEEE  TT      TT  II      II  NNNN  NN  PP      PP
LL      II      88      88  GG      GG  EEEEEEEEE  TT      TT  II      II  NN      NN  PPPPPPPP
LL      II      88      88  GG      GG  EEEEEEEEE  TT      TT  II      II  NN      NN  PPPPPPPP
LL      II      88      88  GG      GG  EEEEEEEEE  TT      TT  II      II  NN      NN  PPPPPPPP
LL      II      88      88  GG      GG  EEEEEEEEE  TT      TT  II      II  NN      NN  PPPPPPPP
LLLLLLLLLLLL  IIIIII  88888888  GGGGGG  EEEEEEEEE  TT      TT  IIIIII  NN      NN  PP      PP
LLLLLLLLLLLL  IIIIII  88888888  GGGGGG  EEEEEEEEE  TT      TT  IIIIII  NN      NN  PP      PP

```



```

LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLLLL  IIIIII  SSSSSSSS

```

```
1 0001 0 MODULE LIB$GET_INPUT (      ! Library $GET on device SYSS$INPUT
2 0002 0
3 0003 0      IDENT = '1-015'          ! File: LIBGETINP.B32  Edit: STAN1015
4 0004 0
5 0005 0      ) =
6 0006 1 BEGIN
7 0007 1
8 0008 1 *****
9 0009 1 *
10 0010 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
11 0011 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
12 0012 1 *  ALL RIGHTS RESERVED.
13 0013 1 *
14 0014 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
15 0015 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
16 0016 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
17 0017 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
18 0018 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
19 0019 1 *  TRANSFERRED.
20 0020 1 *
21 0021 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
22 0022 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
23 0023 1 *  CORPORATION.
24 0024 1 *
25 0025 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
26 0026 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
27 0027 1 *
28 0028 1 *****
29 0029 1
30 0030 1
31 0031 1
32 0032 1 ++
33 0033 1 FACILITY: General Utility Library
34 0034 1
35 0035 1 ABSTRACT:
36 0036 1
37 0037 1      Output a string as a record on device SYSS$INPUT.
38 0038 1
39 0039 1 ENVIRONMENT: User Mode - AST re-entrant
40 0040 1
41 0041 1 AUTHOR: Thomas N. Hastings, CREATION DATE: 8-Aug-1977
42 0042 1
43 0043 1 MODIFIED BY:
44 0044 1
45 0045 1      Thomas N. Hastings, 8-Aug-1977: VERSION 0
46 0046 1 01 - original
47 0047 1 04 - change to SYSS$INPUT
48 0048 1 05 - change to do OPEN at first time
49 0049 1 06 - change to set up RAB for GET_STRING
50 0050 1 0-7 - fix comment
51 0051 1 0-10 - Change to STARLET library. DGP 20-Apr-78
52 0052 1 0-11 - Remove EXTERNAL RMS$RTB. TNH 24-Apr-78
53 0053 1 0-12 - Change REQUIRE files for VAX system build. DGP 28-Apr-78
54 0054 1 0-13 - Change STARLET to RTLSTARLE to avoid conflicts. DGP 1-May-78
55 0055 1 0-14 - Add LIB$GET_COMMAND entry point. TNH 17-June-78
56 0056 1      For now, just copy entire routine.
57 0057 1 0-15 - Make wait if record stream active so AST re-entrant.
```



```
58 0058 1 | Also allocate dynamic string if passed. TNH 29-July-78
59 0059 1 | 0-18 - Make common routine. TNH 29-July-78
60 0060 1 | 0-19 - Use LIB$COPY_R_DX, not DD. TNH 2-Aug-78
61 0061 1 | 0-20 - Change file name to LIBGETINP.B32, and change the name of
62 0062 1 | the REQUIRE file similarly. JBS 14-NOV-78
63 0063 1 | 1-001 - Update version number and copyright notice. JBS 16-NOV-78
64 0064 1 | 1-002 - Declare NULLPARAMETER for new BLISS compiler. JBS 22-NOV-78
65 0065 1 | 1-003 - Change REQUIRE file names from FOR... to OTS... JBS 07-DEC-78
66 0066 1 | 1-003 - Put in extra RETURN to avoid INFO message about a null
67 0067 1 | expression in a value-required context. JBS 22-NOV-78
68 0068 1 | 1-004 - Change LIB$$ to STR$. JBS 23-MAY-1979
69 0069 1 | 1-005 - Change call to STR$COPY. JBS 16-JUL-1979
70 0070 1 | 1-006 - Optionally return the number of characters in the record, so
71 0071 1 | callers with fixed strings can ignore trailing blanks.
72 0072 1 | JBS 06-SEP-1979
73 0073 1 | 1-007 - Revise edit 006 to not return more than the number of bytes
74 0074 1 | requested, and return as a word. This is similar to system
75 0075 1 | services. JBS 18-SEP-1979
76 0076 1 | 1-008 - Use LIB$COPY_R_DX to copy string since STR$COPY_R signals
77 0077 1 | errors.
78 0078 1 | Do string copy even if $GET fails because the string may have
79 0079 1 | been returned. When waiting for record stream to become
80 0080 1 | inactive, do $GET's, not $PUT's! SBL 22-Jan-1980
81 0081 1 | 1-009 - Enhance to recognize additional classes of string descriptors
82 0082 1 | by invoking LIB$ANALYZE_SDESC_R3 to extract length and address
83 0083 1 | of 1st data byte from descriptor. RKR 27-MAY-1981.
84 0084 1 | 1-010 - Correct bugs caused by fact that LIB$ANALYZE_SDESC_R3 returns
85 0085 1 | a word length rather than a byte or longword. SBL 4-Sep-1981
86 0086 1 | 1-011 - Correct comment regarding statuses returned.
87 0087 1 | Add special-case code for string descriptors that "read" like
88 0088 1 | fixed string descriptors to avoid calls to
89 0089 1 | LIB$ANALYZE_SDESC_R3. RKR 7-OCT-1981
90 0090 1 | 1-012 - Redirect jsb's from LIB$ANALYZE_SDESC_R3 to
91 0091 1 | LIB$ANALYZE_SDESC_R2. Use LIB$COPY_R_DX6 to do copying.
92 0092 1 | RKR 18-NOV-1981.
93 0093 1 | 1-013 - Add support for class S0 string descriptors. DG 3-Oct-1983.
94 0094 1 | 1-014 - Change class S0 string descriptors to SB. DG 27-Feb-1984
95 0095 1 | 1-015 - If called with a dynamic string descriptor already containing
96 0096 1 | more than 256 bytes of buffer, use that buffer. STAN 8-Jul-1984.
97 0097 1 | --
98 0098 1 | <BLF/PAGE>
```

```
100 0099 1 |
101 0100 1 | SWITCHES
102 0101 1 |
103 0102 1 |
104 0103 1 | SWITCHES ADDRESSING MODE
105 0104 1 | (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
106 0105 1 |
107 0106 1 |
108 0107 1 | LINKAGES
109 0108 1 |
110 0109 1 | REQUIRE 'RTLIN:STRLNK'; ! Linkage for LIB$ANALYZE_SDESC_R2
111 0294 1 |
112 0295 1 |
113 0296 1 | TABLE OF CONTENTS:
114 0297 1 |
115 0298 1 |
116 0299 1 | FORWARD ROUTINE
117 0300 1 | LIB$GET_INPUT, ! Get string from device SYS$INPUT
118 0301 1 | LIB$GET_COMMAND, ! Get string from device SYS$COMMAND
119 0302 1 | DO_GET; ! Common rout. to do main part of above.
120 0303 1 |
121 0304 1 |
122 0305 1 | INCLUDE FILES:
123 0306 1 |
124 0307 1 |
125 0308 1 | REQUIRE 'RTLIN:RTLPSECT'; ! Define DECLARE_PSECTS macro
126 0403 1 |
127 0404 1 | LIBRARY 'RTLSTARLE'; ! STARLET library for macros and symbols
128 0405 1 |
129 0406 1 |
130 0407 1 | MACROS:
131 0408 1 |
132 0409 1 |
133 0410 1 | EQUATED SYMBOLS:
134 0411 1 |
135 0412 1 |
136 0413 1 | LITERAL
137 0414 1 |
138 0415 1 | K_DYN_STR_MAX = 256; ! Max. size of dynamic string which can
139 0416 1 | ! be handled before truncation
140 0417 1 |
141 0418 1 | PSECT DECLARATIONS:
142 0419 1 |
143 0420 1 | DECLARE_PSECTS (LIB); ! declare PSECTs for LIB$ facility
144 0421 1 |
145 0422 1 | OWN STORAGE:
146 0423 1 |
147 0424 1 |
148 0425 1 | OWN
149 0426 1 | SYS_INPUT_ISI : WORD INITIAL (0); ! ISI for SYS$INPUT
150 0427 1 | SYS_COMMAND_ISI : WORD INITIAL (0); ! ISI for SYS$COMMAND
151 0428 1 |
152 0429 1 |
153 0430 1 | EXTERNAL REFERENCES:
154 0431 1 |
155 0432 1 |
156 0433 1 | EXTERNAL ROUTINE
```

LIB\$GET_INPUT
1-015

1 8
16-Sep-1984 01:00:46 VAX-11 Bliss-32 V4.0-742
14-Sep-1984 12:38:58 [LIBRTL.SRC]LIBGETINP.B32;1

Page 4
(2)

```
: 157      0434 1      LIB$ANALYZE_SDESC_R2 : LIB$ANALYZE_SDESC_JSB_LINK, ! Extract length
: 158      0435 1                                         ! and address of
: 159      0436 1                                         ! 1st data byte
: 160      0437 1                                         ! from descriptor
: 161      0438 1      LIB$COPY_R_DX6 : STRING_JSB ;      ! Copy to any class string
: 162      0439 1
: 163      0440 1      EXTERNAL
: 164      0441 1      LIB$FATERRLIB,      ! LIB -- FATAL ERROR IN LIBRARY
: 165      0442 1      LIB$INPSTRTRU;      ! LIB -- INPUT STRING TRUNCATED
: 166      0443 1
```



```
168 0444 1 GLOBAL ROUTINE LIB$GET_INPUT (      ! Input string from SYS$INPUT
169 0445 1
170 0446 1     GET_STRING,      ! Adr. of string descriptor
171 0447 1     PROMPT_STRING, ! Adr. of optional PROMPT_STRING string
172 0448 1     descriptor
173 0449 1     OUTLEN        ! Optional number of bytes returned
174 0450 1
175 0451 1     ) = ! Value returned is RMS completion code
176 0452 1
177 0453 1 ++
178 0454 1 FUNCTIONAL DESCRIPTION:
179 0455 1
180 0456 1     A line from the current controlling input device, SYS$INPUT, is
181 0457 1     obtained. If an optional PROMPT_STRING is given, output will
182 0458 1     appear on the device, SYS$INPUT, if the device is a terminal;
183 0459 1     otherwise the PROMPT_STRING is ignored. No CRLF is
184 0460 1     appended to the record obtained from RMS. On first call,
185 0461 1     device SYS$INPUT is opened.
186 0462 1     Thus the user can assign the logical name SYS$INPUT to any file
187 0463 1     name in order to redirect I/O.
188 0464 1
189 0465 1 CALLING SEQUENCE:
190 0466 1
191 0467 1     RET_STATUS.wlc.v = LIB$GET_INPUT (get_string.wt.dx
192 0468 1     [,prompt_string.rt.dx
193 0469 1     [,outlen.wv.r]])
194 0470 1
195 0471 1 INPUT PARAMETERS:
196 0472 1
197 0473 1     prompt_string is the address of a string descriptor specifying
198 0474 1     an optional prompt which is output to the
199 0475 1     controlling input device. Where other conventions
200 0476 1     are not established, it is recommended for
201 0477 1     consistency to make prompts be an English word
202 0478 1     followed by a colon(:), one (1) space, and no
203 0479 1     CRLF.
204 0480 1
205 0481 1 OUTPUT PARAMETERS:
206 0482 1
207 0483 1     get_string is the address of string descriptor of any type
208 0484 1     of descriptor supported by LIB$ANALYZE_SDESC.
209 0485 1
210 0486 1     outlen     Is the number of characters placed in the string.
211 0487 1
212 0488 1 IMPLICIT INPUTS:
213 0489 1
214 0490 1     SYS_INPUT_ISI Set on first call to RMS internal stream
215 0491 1     identifier.
216 0492 1
217 0493 1 IMPLICIT OUTPUTS:
218 0494 1
219 0495 1     SYS_INPUT_ISI Set to RMS internal stream identifier
220 0496 1     on first call when SYS$INPUT is OPENed.
221 0497 1
222 0498 1
223 0499 1 COMPLETION STATUS:
224 0500 1
```

```
225 0501 1 1 SS$NORMAL if success.
226 0502 1 1
227 0503 1 1 LIB$INPSTRTRU if input string is bigger than the caller's
228 0504 1 1 fixed length string.
229 0505 1 1 LIB$INVSTRDES if the input descriptor's class is not a
230 0506 1 1 recognized string class.
231 0507 1 1 RMS$_xyz if any RMS error.
232 0508 1 1
233 0509 1 1 SIDE EFFECTS:
234 0510 1 1
235 0511 1 1 Opens file SY$$INPUT on first call and remembers ISI for
236 0512 1 1 subsequent calls.
237 0513 1 1
238 0514 1 1
239 0515 1 1 BEGIN
240 0516 1 1
241 0517 1 1 BUILTIN
242 0518 1 1 NULLPARAMETER;
243 0519 1 1
244 0520 1 1 RETURN DO GET (.GET STRING, ! String to return
245 0521 1 1 (IF NULLPARAMETER (2) THEN 0 ELSE .PROMPT_STRING), ! Optional
246 0522 1 1 (IF NULLPARAMETER (3) THEN 0 ELSE .OUTLEN), ! Optional
247 0523 1 1 ! prompt
248 0524 1 1 ! Optional
249 0525 1 1 ! number of
250 0526 1 1 ! bytes returned
251 0527 1 1 SYS_INPUT_ISI, ! internal stream id for SY$$INPUT
252 0528 1 1 9, ! length of SY$$INPUT string
253 0529 1 1 UPLIT ('SY$$INPUT')); ! name to open first time
254 0530 1 1 END; ! End of LIB$GET_INPUT routine
```

```
.TITLE LIB$GET_INPUT
.IDENT \1-015\

.PSECT _LIB$DATA,NOEXE, PIC,2

0000 00000 SYS_INPUT_ISI:
.WORD 0
0000 00002 SYS_COMMAND_ISI:
.WORD 0

.PSECT _LIB$CODE,NOWRT, SHR, PIC,2

00 00 00 54 55 50 4E 49 24 53 59 53 00000 P.AAA: .ASCII \SY$$INPUT\<0><0><0>

.EXTRN LIB$ANALYZE_SDESC_R2
.EXTRN LIB$SCOPY_R_DX6
.EXTRN LIB$_FATERRCIB, LIB$_INPSTRTRU

0000 00000
EF AF 9F 00002
09 DD 00005
03 00000000' EF 9F 00007
6C 91 0000D
05 1F 00010
OC AC D5 00012

.ENTRY LIB$GET_INPUT, Save nothing
PUSHAB P.AAA
PUSHL #9
PUSHAB SYS_INPUT_ISI
CMPB (APT, #3)
BLSSU 1$
TSTL 12(AP)
```

0444
0528
0520
0523

LIB\$GET_INPUT
1-015

L 8
16-Sep-1984 01:00:46
14-Sep-1984 12:38:58

VAX-11 Bliss-32 V4.0-742
[LIBRTL.SRC]LIBGETINP.B32;1

Page 7
(3)

		04	12	00015	BNEQ	2\$	
		7E	D4	00017	CLRL	-(SP)	
		03	11	00019	BRB	3\$	
02	0C	AC	DD	0001B	PUSHL	OUTLEN	
		6C	91	0001E	CMPB	(AP), #2	0521
		05	1F	00021	BLSSU	4\$	
	08	AC	D5	00023	TSTL	8(AP)	
		04	12	00026	BNEQ	5\$	
		7E	D4	00028	CLRL	-(SP)	
		03	11	0002A	BRB	6\$	
	08	AC	DD	0002C	PUSHL	PROMPT_STRING	
	04	AC	DD	0002F	PUSHL	GET_STRING	0520
0000V	CF	06	FB	00032	CALLS	#6, DO_GET	
		04	00037	RET			0530

; Routine Size: 56 bytes, Routine Base: _LIB\$CODE + 000C

```
256 0531 1 GLOBAL ROUTINE LIB$GET_COMMAND (      ! Input string from SYS$COMMAND
257 0532 1
258 0533 1     GET_STRING,      ! Adr. of string descriptor
259 0534 1     PROMPT_STRING, ! Adr of optional PROMPT_STRING
260 0535 1                                ! descriptor
261 0536 1     OUTLEN      ! Number of chars returned
262 0537 1
263 0538 1                                ) = ! Value returned is RMS completion
264 0539 1                                ! code
265 0540 1
266 0541 1 ++
267 0542 1 FUNCTIONAL DESCRIPTION:
268 0543 1
269 0544 1     A line from the current controlling input device, SYS$COMMAND,
270 0545 1     is obtained.  If an optional PROMPT_STRING is given, output
271 0546 1     will appear on the device, SYS$COMMAND, if the device is a
272 0547 1     terminal; otherwise the PROMPT_STRING is ignored.  No CRLF is
273 0548 1     appended to the record obtained from RMS.  On first call,
274 0549 1     device SYS$COMMAND is opened.
275 0550 1     Thus the user can assign the logical name SYS$COMMAND to any
276 0551 1     file name in order to redirect I/O.  Note: Generally
277 0552 1     LIB$GET_INPUT should be used rather than LIB$GET_COMMAND.
278 0553 1     LIB$GET_COMMAND should only be used when the user has indicated
279 0554 1     that the terminal is explicitly wanted when in an indirect file.
280 0555 1     For example, $INQUIRE or /CONFIRM qualifier.
281 0556 1     Normally, SYS$INPUT and SYS$COMMAND are the same file
282 0557 1     (interactive and batch).  It is only when an interactive user
283 0558 1     uses an indirect file that the devices are different
284 0559 1     (SYS$INPUT = indirect file, SYS$COMMAND remaining associated
285 0560 1     with the terminal).
286 0561 1
287 0562 1 CALLING SEQUENCE:
288 0563 1
289 0564 1     RET_STATUS.wlc.v = LIB$GET_COMMAND (get_string.wt.dx
290 0565 1                                     [,prompt_string.rt.dx
291 0566 1                                     [,outlen.ww.r]])
292 0567 1
293 0568 1 INPUT PARAMETERS:
294 0569 1
295 0570 1     prompt_string is the address of a string descriptor specifying
296 0571 1     an optional prompt which is output to the
297 0572 1     controlling input device.  Where other conventions
298 0573 1     are not established, it is recommended for
299 0574 1     consistency to make prompts be an English word
300 0575 1     followed by a colon(:), one (1) space, and no
301 0576 1     CRLF.
302 0577 1
303 0578 1 OUTPUT PARAMETERS:
304 0579 1
305 0580 1     get_string is the address of string descriptor of any type
306 0581 1     (unspecified, static, dynamic, or varying as
307 0582 1     specified by the DSC$B CLASS field) which is to
308 0583 1     receive the string. (See Chapter 2 -- Section on
309 0584 1     passing strings as output parameters for the
310 0585 1     semantics of each string type.)
311 0586 1     outlen is the number of characters returned to the
312 0587 1     caller.
```

```
0588 1
0589 1
0590 1
0591 1
0592 1
0593 1
0594 1
0595 1
0596 1
0597 1
0598 1
0599 1
0600 1
0601 1
0602 1
0603 1
0604 1
0605 1
0606 1
0607 1
0608 1
0609 1
0610 1
0611 1
0612 1
0613 1
0614 1
0615 1
0616 2
0617 2
0618 2
0619 2
0620 2
0621 2
0622 2
0623 2
0624 2
0625 2
0626 2
0627 2
0628 2
0629 2
0630 2
0631 2
0632 1

IMPLICIT INPUTS:
    SYS_COMMAND_ISI Set on first call to RMS internal stream
                    identifier.

IMPLICIT OUTPUTS:
    SYS_COMMAND_ISI Set to RMS internal stream identifier
                    on first call when SYS$COMMAND is OPENed.

COMPLETION STATUS:
    SSS_NORMAL if success.
    LIB$INPSTRTRU if input string is bigger than the caller's
                    fixed length string.
    LIB$INVARG if the input descriptor's class is not a recognized
                    string type.
    RMS$_xyz if any RMS error.

SIDE EFFECTS:
    Opens file SYS$COMMAND on first call and remembers ISI for
    subsequent calls.

--
BEGIN
BUILTIN
    NULLPARAMETER;
RETURN DO GET (.GET STRING, ! String to return
                (IF NULLPARAMETER (2) THEN 0 ELSE .PROMPT_STRING), ! Optional
                                                                ! prompt
                                                                ! string
                (IF NULLPARAMETER (3) THEN 0 ELSE .OUTLEN), ! Optional
                                                                ! number of
                                                                ! chars returned
                SYS_COMMAND_ISI, ! internal stream id for SYS$COMMAND
                11, ! length of SYS$COMMAND string
                UPLIT ('SYS$COMMAND')) ! name to open first time

END; ! End of LIB$GET_COMMAND routine
```

00 44 4E 41 4D 4D 4F 43 24 53 59 53 00044 P.AAB: .ASCII \SYS\$COMMAND\<0>

		0000 0000	.ENTRY	LIB\$GET_COMMAND, Save nothing	: 0531
	EF	AF 9F 00002	PUSHAB	P.AAB	: 0630
		0B DD 00005	PUSHL	#11	: 0621
	00000000	EF 9F 00007	PUSHAB	SYS_COMMAND_ISI	: 0625
03		6C 91 0000D	CMPB	(APT, #3	

LIB\$GET_INPUT
1-015

B 9
16-Sep-1984 01:00:46
14-Sep-1984 12:38:58

VAX-11 Bliss-32 V4.0-742
[LIBRTL.SRC]LIBGETINP.B32;1

Page 10
(4)

		05	1F	00010		BLSSU	1\$		
	0C	AC	D5	00012		TSTL	12(AP)		
		04	12	00015		BNEQ	2\$		
		7E	D4	00017	1\$:	CLRL	-(SP)		
		03	11	00019		BRB	3\$		
	0C	AC	DD	0001B	2\$:	PUSHL	OUTLEN		
02		6C	91	0001E	3\$:	CMPB	(AP), #2		0622
		05	1F	00021		BLSSU	4\$		
	08	AC	D5	00023		TSTL	8(AP)		
		04	12	00026		BNEQ	5\$		
		7E	D4	00028	4\$:	CLRL	-(SP)		
		03	11	0002A		BRB	6\$		
	08	AC	DD	0002C	5\$:	PUSHL	PROMPT_STRING		
	04	AC	DD	0002F	6\$:	PUSHL	GET_STRING		0621
0000V CF		06	FB	00032		CALLS	#6, DO_GET		
		04	00	00037		RET			0632

; Routine Size: 56 bytes, Routine Base: _LIB\$CODE + 0050

```
359 0633 1 ROUTINE DO_GET (      ! Input string from SYS$INPUT or SYS$COMMAND
360 0634 1
361 0635 1     GET_STRING,          ! Adr. of string descriptor
362 0636 1     PROMPT_STRING,    ! Adr. of optional PROMPT_STRING string
363 0637 1                        ! descriptor
364 0638 1     OUTLEN,          ! Number of chars returned to the caller
365 0639 1     GET_ISI,         ! Adr. of ISI word for this file
366 0640 1     DEVICE_NAME_LEN, ! Length of device name string
367 0641 1     DEVICE_NAME     ! Adr. of device name string
368 0642 1
369 0643 1     ) = ! Value returned is RMS completion code
370 0644 1
371 0645 1 ++
372 0646 1 FUNCTIONAL DESCRIPTION:
373 0647 1
374 0648 1     A line from the current controlling input device, DEVICE_NAME,
375 0649 1     is obtained.  If an optional PROMPT_STRING is given, output
376 0650 1     will appear on the device, DEVICE_NAME, if the device is a
377 0651 1     terminal; otherwise the PROMPT_STRING is ignored.  No CRLF is
378 0652 1     appended to the record obtained from RMS.  On first call, device
379 0653 1     DEVICE_NAME is opened.
380 0654 1     Thus the user can assign the logical name DEVICE_NAME to any
381 0655 1     file name in order to redirect I/O.
382 0656 1
383 0657 1 CALLING SEQUENCE:
384 0658 1
385 0659 1     ret_status.wlc.v = DO_GET (get_string.wt.dx,
386 0660 1                          [prompt_string.rt.dx],
387 0661 1                          [outlen.wv.r],
388 0662 1                          get_isi.mw.r,
389 0663 1                          device_name_len.rl.v,
390 0664 1                          device_name.rt.r)
391 0665 1
392 0666 1 INPUT PARAMETERS:
393 0667 1
394 0668 1     prompt_string is the address of a string descriptor specifying
395 0669 1     an optional prompt which is output to the
396 0670 1     controlling input device.  Where other conventions
397 0671 1     are not established, it is recommended for
398 0672 1     consistency to make prompts be an English word
399 0673 1     followed by a colon(:), one (1) space, and no
400 0674 1     CRLF.
401 0675 1
402 0676 1
403 0677 1     get_isi      Set on first call to RMS internal stream
404 0678 1     identifier.
405 0679 1
406 0680 1     device_name_len is the length of the device_name string in
407 0681 1     bytes.
408 0682 1
409 0683 1     device_name  is the adr. of the device name to be opened
410 0684 1     the first time.
411 0685 1
412 0686 1 OUTPUT PARAMETERS:
413 0687 1
414 0688 1     get_string   is the address of the string descriptor
415 0689 1     which is to receive the string.
```

```
416      0690 1      outlen      : Is the number of characters returned to the caller
417      0691 1
418      0692 1      IMPLICIT INPUTS:
419      0693 1
420      0694 1      NONE
421      0695 1
422      0696 1      IMPLICIT OUTPUTS:
423      0697 1
424      0698 1      NONE
425      0699 1
426      0700 1      COMPLETION STATUS:
427      0701 1
428      0702 1      SSS_NORMAL if success.
429      0703 1
430      0704 1      LIB$_INPSTRTRU if input string is bigger than the caller's
431      0705 1      fixed length string.
432      0706 1      LIB$_INVSTRDES if the input descriptor's class is not a
433      0707 1      recognized string class.
434      0708 1      RMS$_xyz if any RMS error.
435      0709 1
436      0710 1      SIDE EFFECTS:
437      0711 1
438      0712 1      Opens file DEVICE_NAME on first call and remembers ISI for
439      0713 1      subsequent calls by storing ISI in get_isi.
440      0714 1      --
441      0715 1
442      0716 2      BEGIN
443      0717 2
444      0718 2      BUILTIN
445      0719 2      NULLPARAMETER;
446      0720 2
447      0721 2      LOCAL
448      0722 2      GET_STRING_LEN: WORD,      ! length of buffer
449      0723 2      GET_STRING_ADDR,      ! addr of buffer
450      0724 2      PROMPT_STRING_LEN: WORD,    ! length of prompt
451      0725 2      string
452      0726 2      PROMPT_STRING_ADDR,      ! addr of prompt string
453      0727 2      GET_STATUS,      ! status from $GET
454      0728 2      RET_STATUS,      ! status from other
455      0729 2      calls
456      0730 2      FAB : $FAB_DECL,      ! FAB
457      0731 2      RAB : $RAB_DECL,      ! RAB
458      0732 2      DYNAMIC_STR_BUF : VECTOR [K_DYN_STR_MAX, BYTE, UNSIGNED];
459      0733 2      ! temporary buffer for
460      0734 2      ! dynamic string case.
461      0735 2
462      0736 2      MAP
463      0737 2      GET_STRING : REF BLOCK [8, BYTE],      ! String descriptor
464      0738 2      PROMPT_STRING : REF BLOCK [8, BYTE],    ! String descriptor
465      0739 2      OUTLEN : REF VECTOR [1, WORD, UNSIGNED], ! Number of characters
466      0740 2      returned to the user
467      0741 2      GET_ISI : REF VECTOR [1, WORD, UNSIGNED]; ! Place to remember
468      0742 2      ! ISI in static
469      0743 2      ! storage
470      0744 2
471      0745 2      IF (.GET_ISI [0] EQL 0)
472      0746 2      THEN
```



```

+ First call, initialize FAB
-
BEGIN
$FAB_INIT (FAB = FAB,
           FAC = GET,      ! file access: GET
           FNA = .DEVICE_NAME, ! file name: DEVICE_NAME
                               (SYS$INPUT or SYS$COMMAND)
           FNS = .DEVICE_NAME_LEN); ! file name size:
                                   ! 9 or 11 bytes

+ Open DEVICE_NAME, remember RMS internal stream identifier
-
RET_STATUS = $OPEN (FAB = FAB);      ! fab addr : FAB

+ If the OPEN fails, return the RMS status code.
-
IF ( NOT .RET_STATUS) THEN RETURN (.RET_STATUS);

$RAB_INIT (FAB = FAB, RAB = RAB);
RET_STATUS = $CONNECT (RAB = RAB); ! connect RAB to the file

+ Similarly, if the CONNECT fails, return the RMS status code.
-
IF ( NOT .RET_STATUS) THEN RETURN (.RET_STATUS);

GET_ISI [0] = .RAB [RAB$W_ISI];      ! remember ISI
END      ! of first call

ELSE

+ file already open, just initialize RAB
+ including internal stream identifier returned from first $OPEN
-
BEGIN      ! file already open
$RAB_INIT (FAB = FAB, RAB = RAB);
RAB [RAB$W_ISI] = .GET_ISI [0];
END;      ! file already open

+ Determine which buffer area to read into, and how long it is.
+ Use LIB$ANALYZE_SDESC_R2 to get length and address of 1st data byte
+ of caller's buffer.
+ If the descriptor is invalid, return status returned by
+ LIB$ANALYZE_SDESC_R2.
-
IF .GET_STRING [DSC$B_CLASS] GTRU DSC$K_CLASS_D
THEN      ! Use generalized extraction
BEGIN
LOCAL RET_STATUS ;
RET_STATUS = LIB$ANALYZE_SDESC_R2 ( .GET_STRING ;
```

```
530 0804 GET_STRING_LEN,  
531 0805 GET_STRING_ADDR ) ;  
532 0806  
533 0807 IF NOT .RET_STATUS THEN RETURN (.RET_STATUS) ;  
534 0808 END  
535 0809  
536 0810 ELSE ! Fetch length and address directly  
537 0811  
538 0812 BEGIN  
539 0813 GET_STRING_LEN = .GET_STRING [DSC$W_LENGTH] ;  
540 0814 GET_STRING_ADDR = .GET_STRING [DSC$A_POINTER] ;  
541 0815 END ;  
542 0816  
543 0817  
544 0818 + If GET_STRING is dynamic, we arrange to read onto a area of the  
545 0819 stack since the dynamic string may not be allocated.  
546 0820 However, if the dynamic string happens to be allocated and if it  
547 0821 contains more space than we would have used (256 bytes), then  
548 0822 we should use the space that the caller has provided.  
549 0823 -  
550 0824 IF .GET_STRING [DSC$B_CLASS] EQL DSC$K_CLASS_D  
551 0825 AND .GET_STRING_LEN LESS K_DYN_STR_MAX  
552 0826 THEN  
553 0827 BEGIN  
554 0828 GET_STRING_LEN = K_DYN_STR_MAX ;  
555 0829 GET_STRING_ADDR = DYNAMIC_STR_BUF ;  
556 0830 END ;  
557 0831  
558 0832 +  
559 0833 If GET_STRING was varying, the length we want is MAXSTRLEN, not  
560 0834 CURLEN as returned by LIB$ANALYZE_SDESC_R2.  
561 0835 -  
562 0836 IF .GET_STRING [DSC$B_CLASS] EQL DSC$K_CLASS_VS  
563 0837 THEN  
564 0838 BEGIN  
565 0839 GET_STRING_LEN = .GET_STRING [DSC$W_MAXSTRLEN] ;  
566 0840 END ;  
567 0841 +  
568 0842 Set up RAB buffer address and length fields based on our computations.  
569 0843 -  
570 0844 RAB [RAB$L_UBF] = .GET_STRING_ADDR ;  
571 0845 RAB [RAB$W_USZ] = .GET_STRING_LEN ;  
572 0846  
573 0847 +  
574 0848 Setup prompt buffer address and size in RAB if PROMPT_STRING string  
575 0849 present. If Prompt string descriptor invalid, return status returned  
576 0850 by LIB$ANALYZE_SDESC_R2.  
577 0851 -  
578 0852  
579 0853 IF ( NOT NULLPARAMETER (2))  
580 0854 THEN  
581 0855 BEGIN  
582 0856 IF .PROMPT_STRING [DSC$B_CLASS] GTRU DSC$K_CLASS_D  
583 0857 THEN ! Use generalized extraction  
584 0858 BEGIN  
585 0859 LOCAL RET_STATUS ;  
586 0860 RET_STATUS = LIB$ANALYZE_SDESC_R2 ( .PROMPT_STRING ;
```

```
587 0861 4
588 0862 4
589 0863 4
590 0864 4
591 0865 4
592 0866 3
593 0867 4
594 0868 4
595 0869 4
596 0870 4
597 0871 4
598 0872 4
599 0873 4
600 0874 4
601 0875 4
602 0876 4
603 0877 4
604 0878 4
605 0879 4
606 0880 4
607 0881 4
608 0882 4
609 0883 4
610 0884 4
611 0885 4
612 0886 4
613 0887 4
614 0888 4
615 0889 4
616 0890 4
617 0891 4
618 0892 4
619 0893 4
620 0894 4
621 0895 4
622 0896 4
623 0897 4
624 0898 4
625 0899 4
626 0900 4
627 0901 4
628 0902 4
629 0903 4
630 0904 4
631 0905 4
632 0906 4
633 0907 4
634 0908 4
635 0909 4
636 0910 4
637 0911 4
638 0912 4
639 0913 4
640 0914 4
641 0915 4
642 0916 4
643 0917 4

PROMPT_STRING_LEN,
RAB [RAB$$_PBF] );!addr.
IF NOT .RET_STATUS THEN RETURN (.RET_STATUS) ;
END

ELSE ! Fetch length and address directly
BEGIN
PROMPT_STRING_LEN = .PROMPT_STRING [DSC$$_LENGTH] ;
RAB [RAB$$_PBF] = .PROMPT_STRING [DSC$$_POINTER] ;
END;

RAB [RAB$$_PSZ] = MINU (255, .PROMPT_STRING_LEN);
RAB [RAB$$_PMT] = 1;
END;

+
Input the string as a single record
Return RMS error status if not RECORD TOO BIG or RECORD STREAM ACTIVE.
On record stream active, wait and try again.
-
GET_STATUS = $GET (RAB = RAB);
IF NOT .GET_STATUS
THEN
BEGIN
WHILE (.RAB [RAB$$_STS] EQL RMS$$_RSA) DO
BEGIN
$WAIT (RAB = RAB);
GET_STATUS = $GET (RAB = RAB);
END;
END;

+
Having read the record, we now have to worry about the semantics of
GET_STRING.
If GET_STRING has fixed-length semantics, we must blank fill the tail
end of the buffer that RMS didn't fill.
If GET_STRING has dynamic semantics, the input got read into an area
on the stack (or in the user's buffer) and needs to be copied
to GET_STRING.
If GET_STRING has varying string semantics we need to adjust the
CURLEN field to reflect how many bytes it really contains.
-
CASE .GET_STRING [DSC$$_CLASS]
FROM DSC$$_CLASS_2 TO DSC$$_CLASS_SB OF
SET
+
Classes with fixed-length string semantics
-
[DSC$$_CLASS_2, | Unspecified
DSC$$_CLASS_3, | Scalar
DSC$$_CLASS_4, | Array
DSC$$_CLASS_5, | Scaled decimal
DSC$$_CLASS_6, | Non-contiguous array
DSC$$_CLASS_7, | String with bounds
DSC$$_CLASS_8, |
BEGIN ! fixed length processing
!+
```



```
644 0918      ! Because we opened the file in MOVE mode and used the
645 0919      ! caller's string as the UBF, we need only blank pad the
646 0920      ! area beyond the string; the actual data has been
647 0921      ! moved into the front of the user's string by RMS.
648 0922
649 0923      CHSFILL (XC' ',
650 0924      .GET_STRING_LEN - .RAB [RAB$W_RSZ],
651 0925      .GET_STRING_ADDR + .RAB [RAB$W_RSZ]);
652 0926      RET_STATUS = T;      ! To denote copy success
653 0927      END;      ! fixed length processing
654 0928
655 0929
656 0930      !+
657 0931      ! Classes with varying string semantics
658 0932      [DSC$K_CLASS_VS]:      ! Varying string
659 0933      BEGIN      ! varying length processing
660 0934      (.GET_STRING [DSC$A_POINTER]) < 0, 16 > = .RAB [RAB$W_RSZ];
661 0935      ! CURLen <- bytes gotten
662 0936      RET_STATUS = 1;      ! To denote copy success
663 0937      END;      ! varying length processing
664 0938
665 0939
666 0940      !+
667 0941      ! Classes with dynamic string semantics
668 0942      ! Even if we had read into the user's buffer, we still must
669 0943      ! ensure that the length is correct.
670 0944
671 0945      [DSC$K_CLASS_D]:      ! Dynamic string
672 0946      BEGIN      ! dynamic length processing
673 0947      RET_STATUS = LIB$COPY_R_DX6 (.RAB [RAB$W_RSZ],
674 0948      (IF .GET_STRING_LEN LSSU K_DYN_STR_MAX
675 0949      THEN
676 0950      DYNAMIC_STR_BUF
677 0951      ELSE
678 0952      .GET_STRING_ADDR),
679 0953      .GET_STRING);
680 0954      END;      ! dynamic length processing
681 0955
682 0956      [INRANGE, OUTRANGE]:      ! Should never take this path since
683 0957      ! a bad descriptor class code should
684 0958      ! have gotten caught the first time
685 0959      ! we tried to get GET_STRING's length
686 0960      ! and address.
687 0961      RETURN (LIB$FATERRLIB);
688 0962
689 0963      TES;
690 0964      !+
691 0965      ! If requested, tell the caller the number of bytes actually returned,
692 0966      ! not counting blank padding, if any.
693 0967
694 0968
695 0969      IF ( NOT NULLPARAMETER (3))
696 0970      THEN OUTLEN [0] = MINU (.RAB [RAB$W_RSZ], .GET_STRING_LEN);
697 0971
698 0972      !+
699 0973      ! Return proper status code.
700 0974
```

[illegible]

! Record too big

```
! End of routine DO_GET
```

```
.EXTRN  SYS$OPEN, SYS$CONNECT
.EXTRN  SYS$GET,  SYS$WAIT
```

PC	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419
----	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

LIB
1-0

SPECIAL

54		52	D0	000AB	MOVL	R2, R4	:	
57		51	D0	000AB	MOVL	R1, GET_STRING_LEN	:	
08		50	E8	000AE	BLBS	RET_STATUS, 6\$:	0807
			04	000B1	RET		:	
57		66	B0	000B2	5\$: MOVW	(R6), GET_STRING_LEN	:	0813
54	04	A6	D0	000B5	MOVL	4(R6), GET_STRING_ADDR	:	0814
02	03	A6	91	000B9	6\$: CMPB	3(R6), #2	:	0824
		0F	12	000BD	BNEQ	7\$:	
0100	8F	57	B1	000BF	CMPW	GET_STRING_LEN, #256	:	0825
		08	1E	000C4	BGEQU	7\$:	
57	0100	8F	B0	000C6	MOVW	#256, GET_STRING_LEN	:	0828
54		6E	9E	000CB	MOVAB	DYNAMIC_STR_BUF, GET_STRING_ADDR	:	0829
0B	03	A6	91	000CE	7\$: CMPB	3(R6), #11	:	0836
		03	12	000D2	BNEQ	8\$:	
57		66	B0	000D4	MOVW	(R6), GET_STRING_LEN	:	0839
90	AD	54	D0	000D7	8\$: MOVL	GET_STRING_ADDR, RAB+36	:	0844
8C	AD	57	B0	000DB	MOVW	GET_STRING_LEN, RAB+32	:	0845
	02	6C	91	000DF	CMPB	(APT, #2	:	0853
		3C	1F	000E2	BLSSU	12\$:	
		08	AC	D5	000E4	TSTL	:	
		37	13	000E7	BEQL	12\$:	
53	08	AC	D0	000E9	MOVL	PROMPT_STRING, R3	:	0856
02	03	A3	91	000ED	CMPB	3(R3), #2	:	
		0D	1B	000F1	BLEQU	9\$:	
50		53	D0	000F3	MOVL	R3, R0	:	0860
		6A	16	000F6	JSB	LIB\$ANALYZE_SDESC_R2	:	
9C	AD	52	D0	000F8	MOVL	R2, RAB+48	:	0862
	09	50	E8	000FC	BLBS	RET_STATUS, 10\$:	0863
			04	000FF	RET		:	
51		63	B0	00100	9\$: MOVW	(R3), PROMPT_STRING_LEN	:	0868
9C	AD	A3	D0	00103	MOVL	4(R3), RAB+48	:	0869
50		51	3C	00108	10\$: MOVZWL	PROMPT_STRING_LEN, R0	:	0872
00FF	8F	50	B1	0010B	CMPW	R0, #255	:	
		04	1B	00110	BLEQU	11\$:	
50	FF	8F	9A	00112	MOVZBL	#255, R0	:	
AD		50	90	00116	11\$: MOVB	R0, RAB+52	:	
FF73	CD	40	8F	0011A	BISB2	#64, RAB+7	:	0873
		FF6C	CD	9F	00120	12\$: PUSHAB	:	0881
6B		01	FB	00124	CALLS	#1, SYSS\$GET	:	
58		50	D0	00127	MOVL	R0, GET_STATUS	:	
22		58	E8	0012A	BLBS	GET_STATUS, 14\$:	0883
000182DA	8F	FF74	CD	D1	0012D	13\$: CMPL	:	0886
			17	12	00136	BNEQ	:	
		FF6C	CD	9F	00138	PUSHAB	:	0888
00000000G	00	FF6C	01	FB	0013C	CALLS	:	0889
		FF6C	CD	9F	00143	PUSHAB	:	
			01	FB	00147	CALLS	:	
6B		50	D0	0014A	MOVL	R0, GET_STATUS	:	
58		DE	11	0014D	BRB	13\$:	0886
	0F	00	A6	8F	0014F	14\$: CASEB	:	0904
0020	0045	0028	0028	00154	15\$: .WORD	17\$-15\$,-	:	
0020	0020	0020	0028	0015C		17\$-15\$,-	:	
003B	0028	0028	0020	00164		20\$-15\$,-	:	
0028	0020	0020	0020	0016C		16\$-15\$,-	:	
						17\$-15\$,-	:	
						16\$-15\$,-	:	
						16\$-15\$,-	:	

51

20

000181A8 OC

Address	Hex	Op	OpHex	OpDec	Inst	Comment	Label	
50	00000000G	00	9E	00174	16\$:	MOVAB	LIB\$_FATERRLIB, R0	0962
			04	0017B		RET		
50	8E	AD	3C	0017C	17\$:	MOVZWL	RAB+34, R0	0924
51		57	3C	00180		MOVZWL	GET_STRING_LEN, R1	
51		50	C2	00183		SUBL2	R0, R1	
6E		00	2C	00186		MOVCS	#0, (SP), #32, R1, (R0)[GET_STRING_ADDR]	0925
		6044		0018B				
		05	11	0018D		BRB	19\$	0926
B6	8E	AD	B0	0018F	18\$:	MOVW	RAB+34, @4(R6)	0935
59		01	D0	00194	19\$:	MOVL	#1, RET_STATUS	0937
		22	11	00197		BRB	23\$	0904
8F		57	B1	00199	20\$:	CMPW	GET_STRING_LEN, #256	0949
		08	1E	0019E		BGEQU	21\$	
52		6E	9E	001A0		MOVAB	DYNAMIC_STR_BUF, R2	
51		52	C0	001A3		MOVL	R2, R1	
		03	11	001A6		BRB	22\$	
51		54	D0	001A8	21\$:	MOVL	GET_STRING_ADDR, R1	0953
52		56	D0	001AB	22\$:	MOVL	R6, R2	0948
50	8E	AD	3C	001AE		MOVZWL	RAB+34, R0	
	00000000G	00	16	001B2		JSB	LIB\$SCOPY R DX6	
59		50	D0	001B8		MOVL	R0, RET_STATUS	
03		6C	91	001BB	23\$:	CMPB	(AP), #3	0969
		15	1F	001BE		BLSSU	25\$	
	0C	AC	D5	001C0		TSTL	12(AP)	
		10	13	001C3		BEQL	25\$	
50	8E	AD	3C	001C5		MOVZWL	RAB+34, R0	0970
50		57	B1	001C9		CMPW	GET_STRING_LEN, R0	
		03	1E	001CC		BGEQU	24\$	
50		57	3C	001CE		MOVZWL	GET_STRING_LEN, R0	
BC		50	B0	001D1	24\$:	MOVW	R0, @OUTLEN	
8F		58	D1	001D5	25\$:	CMPB	GET_STATUS, #98728	0976
		08	12	001DC		BNEQ	26\$	
50	00000000G	00	9E	001DE		MOVAB	LIB\$_INPSTRTRU, R0	0978
			04	001E5		RET		0979
04		58	E8	001E6	26\$:	BLBS	GET_STATUS, 27\$	
50		58	D0	001E9		MOVL	GET_STATUS, R0	0981
			04	001EC		RET		
04		59	E8	001ED	27\$:	BLBS	RET_STATUS, 29\$	0982
50		59	D0	001F0	28\$:	MOVL	RET_STATUS, R0	0984
			04	001F3		RET		
50		01	D0	001F4	29\$:	MOVL	#1, R0	0985
			04	001F7		RET		0987

```
; Routine Size: 504 bytes,    Routine Base: _LIB$CODE + 0088
```

```

: 714      0988 1 END
: 715      0989 1

```

```
!End of module LIB$GET_INPUT
```

LIB\$GET_INPUT
1-015

L 9
16-Sep-1984 01:00:46
14-Sep-1984 12:38:58

VAX-11 Bliss-32 V4.0-742
[LIBRTL.SRC]LIBGETINP.B32;1

Page 20
(5)

: 716 0990 0 ELUDOM

PSECT SUMMARY

Name	Bytes	Attributes
LIB\$DATA	4	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, PIC, ALIGN(2)
LIB\$CODE	640	NOVEC, NOWRT, RD, EXE, SHR, LCL, REL, CON, PIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	87	0	581	00:00.8

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS\$:LIBGETINP/OBJ=OBJ\$:LIBGETINP MSRC\$:LIBGETINP/UPDATE=(ENH\$:LIBGETINP)

: Size: 616 code + 28 data bytes
: Run Time: 00:11.7
: Elapsed Time: 00:41.5
: Lines/CPU Min: 5089
: Lexemes/CPU-Min: 40128
: Memory Used: 192 pages
: Compilation Complete

0207 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

LIBFLTUND
LIS

LIBGETSYI
LIS

LIBICHAR
LIS

LIBINITIA
LIS

LIBEXUPF
LIS

LIBGETFOR
LIS

LIBGETINP
LIS

LIBINISHR
LIS

LIBGETDVI
LIS

LIBGETOPC
LIS

LIBFNDING
LIS

LIBGETMSG
LIS

LIBINDEX
LIS

LIBINSQHI
LIS

LIBGETUP1
LIS

LIBGETTAB
LIS